EE2703 : Applied Programming Lab
Assignment 3
Fitting Data to Models

Sai Shashank GP
EE20B040

February 18, 2022

0.1 Introduction

In this assignment, we are going to deal with the noisy data by study it's characteristics and trying to find a good approximation to it. We are going to use **pylab** for plotting graphs, performing scientific calculations and mimick some standard functions .

0.2 Assignment

0.2.1 Prerequisites

Importing the standard libraries

```
from pylab import *
from scipy.linalg import lstsq
import scipy.special as sp
import numpy as np
from sklearn.metrics import mean_squared_error as mse

global N, k
N = 101
k = 9
```

0.2.2 Q1

Creating the data. For this I'm going to define a function which completes the task of genrating the noisy data

```
def Q1():
    '''This function executes the task in Q1'''

    t=linspace(0,10,N)
    y=1.05*sp.jn(2,t)-0.105*t
    Y=meshgrid(y,ones(k),indexing='ij')[0]
    global scl
    scl=logspace(-1,-3,k)
    n=dot(randn(N,k),diag(scl))
    yy=Y+n

    savetxt("fitting.dat",c_[t,yy])
    print('Data file is created. Do you want to load it into the program? Type Q2')
    return
```

0.2.3 Q2

After the data being created, I'm defining a function to load the data into the program

```
def Q2():
    '''This function executes the task in Q2'''
    global fileArray, time, data
```

```
    try:
        fileArray = np.loadtxt('fitting.dat')
    except FileNotFoundError:
        print('ERROR: File not created. Type Q1')
    time, data = fileArray[:, 0], fileArray[:, 1:]
    return
```

0.2.4 Q3

Now we plot the function for various noise amounts.

```
def Q3():
    '''This function executes the task in Q3'''
    Q2()
    Legend = list(np.round_(scl, decimals=3))

    figure(0)
    plot(time, data)
    xlabel(r'$t$', size=20)
    ylabel(r'$f(t) + n$', size=20)
    title(r'Figure 0')
    legend(Legend)
    show()
```

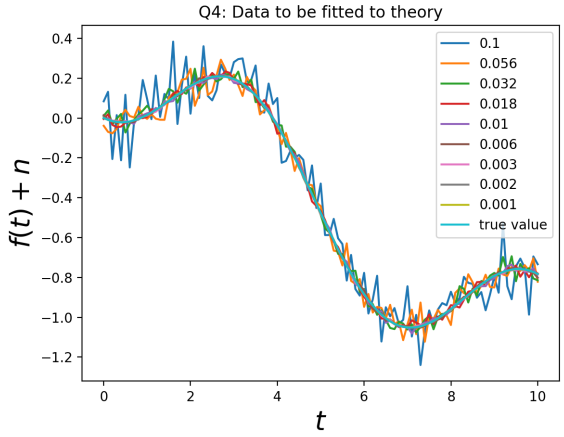

Figure 1: Noisy data along with True data

0.2.5 Q4

This time, we add the true value graph to the previous graph

```
def Q4():
    Q2()
    Legend = list(np.round_(scl, decimals=3))
    Legend.append('true value')
    data_ = np.hstack((data, np.reshape(np.transpose(g(time, 1.05, -0.105)), (N, 1))))
```

```
    plot(time, data_)
    xlabel(r'$t$', size=20)
    ylabel(r'$f(t) + n$', size=20)
    title(r'Q4: Data to be fitted to theory')
    legend(Legend, loc=1)
    show()
```

And that true function array is created by this following function

```
def g(t, A, B):
    return A*sp.jn(2, t) + B*t
```

0.2.6 Q5

To better interpret the data, I'm trying to plot the data's error bars. I'll plot for the first column

```
def Q5():
    '''This function executes the task Q5'''
    Q2()
    plot(time, g(time, 1.05, -0.105))
    errorbar(time[::5], data[:, 0][::5], scl[0], fmt='ro')
    grid(True)
    legend(['f(t)', 'errorbar'])
    title(r'Q5: Data points for stdev=0.1 along with true function')
    show()
```

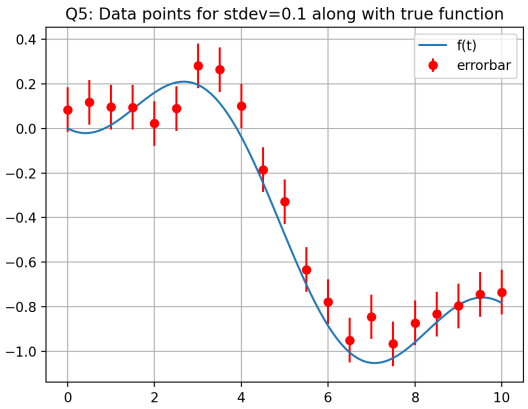

Figure 2: Error bar graph with $\sigma_n = 0.1$

0.2.7 Q6

Since the actual function given to us before adding noise is $1.05J_2(t) - 0.105t$, we can try and find the true value array using Matrix Multiplication as follows

```
def Q6():
    '''This function validates the statement that matrix (J(t) t)*(A; B) = g(t, A, B)'''
    Q2()
    global M, p, g_
    list_M = [[sp.jn(2, i), i] for i in time]
    M = np.array(list_M)
    p = np.array(np.reshape(np.array([1.05, -0.105]), (2, )))
    g_ = np.dot(M, p)
    return np.array(g_ == g(time, 1.05, -0.105)).all()
```

0.2.8 Q7

By the above idea, we can do a little experiment here. I'm going to consider a range of values for A and B which are around the actual values.

```
def Q7():
    '''This function executes the task in Q7'''
    Q2()
    global E, A, B
    A = list(np.linspace(0, 2, 21))
    B = list(np.linspace(-0.2, 0, 21))
    E = np.zeros((21, 21))
    for i in range(len(A)):
        for j in range(len(B)):
            E[i][j] = mse(g(time, A[i], B[j]), data[:, 0])
```

0.2.9 Q8

Now, I'll plot the contour plot of the previously created error matrix

```
def Q8():
    '''This function executes the task in Q8'''
    Q7()
    plot(1.05, -0.105, 'ro')
    annotate('Exact Value', (1.05, -0.105))
    contour(np.array(A), np.array(B), E)
    xlabel('A')
    ylabel('B')
    title(r'Q8: Contour plot of e_ij')
    show()
```

0.2.10 Q9

Using the least square fitting method, we can estimate the A and B values for a noisy graph. First, let's do it for the actual true value and check if we are getting the expected output of A and B

```
def Q9():
    _ = Q6()
```

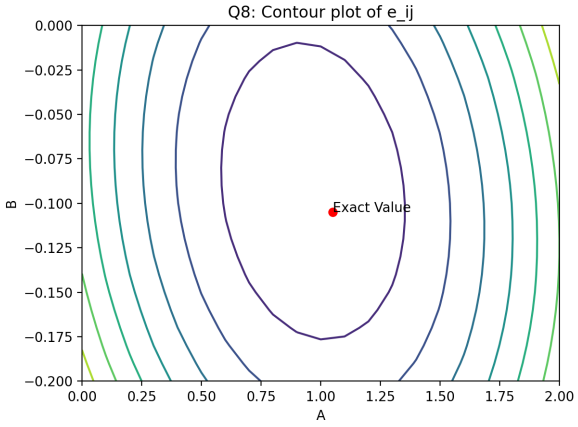

Figure 3: ϵ_{ij} contour plot

```
    ABarr, *l = lstsq(M, g_)
    print(ABarr)
```

0.2.11 Q10

Now, let's apply the above mentioned method for the noisy data we have and plot the error trend with noise σ_n

```
def Q10():
    '''This function executes the task in Q10'''
    _ = Q6()
    Aerr_ = []
    Berr_ = []
    MSE = []
    for i in range(k):
        a, mse, *b= lstsq(M, data[:, i])
        Aerr_.append(abs(1.05-a[0]))
        Berr_.append(abs(-0.105-a[1]))
        MSE.append(mse)
    Aerr = np.array(Aerr_)
    Berr = np.array(Berr_)
    MSE = np.array(MSE)
    plot(scl, Aerr, 'ro--')
    plot(scl, Berr, 'bo--')
    plot(scl, MSE)
    legend(['Aerr', 'Berr', 'LMSE'])
    grid(True)
    title(r'Q10: Variation of error with noise')
    show()
```

0.2.12 Q11

As mentioned in the question, let's plot the error trend in log log scale

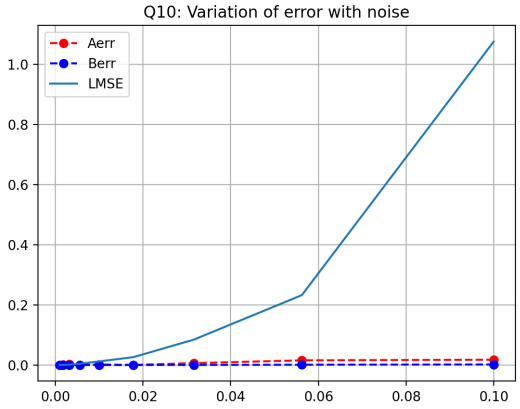

Figure 4: Error trend wrt noise

```
def Q11():
    '''This function executes the task in Q11'''
    _ = Q6()
    Aerr_ = []
    Berr_ = []
    MSE = []
    for i in range(k):
        M_ = np.c_[data[:, i], time]
        a, mse, *b= lstsq(M_, g_)
        Aerr_.append(abs(1.05-a[0]))
        Berr_.append(abs(-0.105-a[1]))
        MSE.append(mse)
    Aerr = np.array(Aerr_)
    Berr = np.array(Berr_)
    MSE = np.array(MSE)
    loglog(scl, Aerr, 'ro--')
    loglog(scl, Berr, 'bo--')
    loglog(scl, MSE)
    legend(['Aerr', 'Berr', 'LMSE'])
    grid(True)
    title(r'Q11: Variation of error with noise in loglog scale')
    show()
```

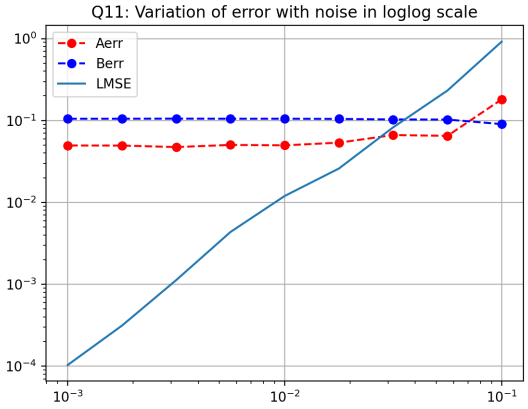

Figure 5: Error trend wrt noise in log log scale

0.3 Conclusions

- There is only one minimum and it is close to the exact value
- The variation of Error with Noise is non linear but increasing with noise in linear scale
- The variation of Error with Noise is linear (approximately) with Noise